Name : Hardik Pandey UID : 21BCS 7892

Section / Grp : 21BCS - 5 - B Subject : NMOUP (21CSH - 459)

ASSIGNMENT

Q1. Unconstrained Optimization Problem :

Develop and implement a gradient descent algorithm in Python to minimize a multivariable function, such as the Rosenbrock function. Analyze the impact of different learning rates on the convergence rate of the algorithm.

Ans 1. Unconstrained Optimization refers to finding the minimum of a function without any restriction on variable value. A common method to solve such problem is the gradient descent algorithm, which iteratively updates variables in the direction of the negative gradient to reduce the function value.

In this case, we will minimize the Rosenbrock Function, which is commonly used to evaluate the performance of optimization algorithms.

The Rosenbrock function is defined as :

$$f(x, y) = (a-x)^2 + b(y-x^2)^2$$

where, $a = 1$, $b = 1$ are constants.

This function has a global minimum at $x = 1$, $y = 1$, but the path to the minimum is curved and poses challenges for optimization algorithms.

The gradient descent algo updates the values of variables x & y iteratively as :

$$x_{n+1} = x_n - \alpha . \frac{\partial f}{\partial x}$$

$$y_{n+1} = y_n - \alpha . \frac{\partial f}{\partial y}$$

where, α = learning rate

$\frac{\partial f}{\partial x}$ & $\frac{\partial f}{\partial y}$ are partial derivatives of the function wrt x and y.

Thus, partial derivative of Rosenbrock function are :

$$\frac{\partial f}{\partial x} = -2(a-x) - 4b(xy - x^3)$$

$$\frac{\partial f}{\partial y} = 2b(y - x^2)$$

The learning rate α plays a crucial role in the convergence speed and accuracy of the algorithm. A small learning rates results in slow convergence but ensures stability. A large learning rate speeds up convergence but may cause overshooting, preventing algorithm from reaching the minimum.
Hence, choice of the learning rate is crucial for the performance of algorithm. Through this experiment, moderate learning rates provide a balance between speed and accuracy.

Q2. Linear Programming : Formulate and resolve real world resource allocation problem using linear programming. For example, a factory produces two products and each requires a certain amount of material and labour. The factory has limited resources for both. Implement this optimization problem using Python's SciPy or PuLP library.

Ans 2. Linear Programming (LP) is a method used to optimize a linear objective function, subject to linear equality and inequality constraints.

In this scenario, we will formulate a resource allocation problem for a factory that produces two products, say Product A and Product B. Each product requires certain amount of material and labor and the factory has limited resources for both.

Let, x_1 = no. of units of Product A produced

x_2 = no. of units of Product B produced

The goal is to maximize the profit, subject to constraints on the availability of material and labour.

Thus, the implementation using SciPy or PuLP is as follow:

```
from scipy.optimize import linprog

c = [-40, -30]

A = [
        [3,2],    # Material constraint
        [2,4]     # labor constraint
    ]

b = [120, 160]

x-bounds = (0, None)

bounds = [x-bounds, x-bounds]

res = linprog( c, A-ub = A, bounds = bounds,
        method = 'highs')
```

```
print (" Optimization units of Product A : {res.x[0]}")
print (" Optimization units of Product B : {res.x[1]}")
print ( " Maximum Profit : $ {-res.fun }")
```

Hence, using SciPy we obtained the optimal no. of units of Product A and B respectively to produce maximum profit. Using Scipy or PuLP python libraries simplifies formulating and solving such optimization problems, offering flexibility and ease of implementation.

Q3. Quadratic Programming Application : Design a quadratic programming problem where you optimize a cost function representing the total cost of production, subject to a set of constraints on resource usage. Use Python libraries like CVXopt to solve this problem and compare the results with different solver parameters.

Ans 3. Quadratic Programming is an optimization method for problems where the objective function is quadratic and the constraints are linear. In this scenario, we will design a QP problem to minimize the total production cost for two products, subject to constraints on resource usage.

Implementation using cvxopt is as follow:

```
import cvxopt .

Q = cvxopt . matrix ([[ 1.0, 0.0], [0.0, 2.0]])

P = cvxopt. matrix ( [3.0, 4.0])

A = cvxopt. matrix ([ [3.0, 2.0], [2.0, 5.0], [-1.0, 0.0],
     [0.0, -1.0]])

b = cvxopt .matrix ([100.0, 80.0, 0.0, 0.0])

sol = cvxopt. solvers .qp (Q, P, A, b)

print (" Optimal units of Product A : {sol ['x'][0]}")

print (" Optimal units of Product B : {sol ['x'] [1]}")

print (" Minimum Cost : {sol [' primal objective ']}")
```

The cvxopt. solvers.qp function allows

solver parameters like tolerance, maximum iterations and the type of solver (interior point or conic). Quadratic Programming efficiently models production cost optimization with non-linear cost structures for improved performance in different scenarios.

Q4. Simulated Annealing for Optimization:

Solve a combinatorial optimization problem (such as the traveling salesman problem or a knapsack problem) using simulated annealing. Write a Python implementation and evaluate the effect of varying the temperature decay schedule on the quality of final solution.

Ans 4. Simulated Annelia is a probabalistic optimization technique inspired by the annealing process in metallurgy. It is commonly used to solve combinatorial optimization problems such as the knapsack problem, where the objective is to maximize the value of items selected without exceeding the weight capacity of knapsack.

In the 0/1 knapsack problem :

- Each item i has a value v_i and a weight w_i.
- The goal is to maximize the total value while the total weight does not exceed the knapsack capacity.

Let, $x_i = 1$ if item i is selected, otherwise $x_i = 0$.

Therefore,

$$\text{Total Value} = \Sigma v_i x_i$$

$$\text{Total Weight} = \Sigma w_i x_i \leq \text{Capacity}$$

In this solution we explore the solution space by accepting worse solutions with a certain probability, allowing it to escape local optima.

This probability decreases as the temperature decreases over-time. The temperature decay schedule is a key parameter that affects the convergence quality.

Q5. Optimization in Machine Learning:

Implement gradient descent to train a linear regression model. Extend your code to solve the regression problem using Lasso or Ridge regression (which can lee modeled as a constrained optimization problem) and analyze the impact of regularization on model performance.

Ans 5. Creating accurate predictive models is a fundamental task in data analysis. Based on the difference between model prediction and actual data points we try to find the model which give better accuracy on dataset.

In order to find these parameters we apply ~~gradient~~ gradient descent on the cost function of the machine learning model.

Gradient Descent is an iterative optimization algo that tries to find the optimum value of an objective function. The main aim of gradient descent is to find the best parameters of a model which gives the highest accuracy on training as well as testing dataset.

Cost Function :

$$\boxed{J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x_i) - y_i]^2}$$

Gradient Descent :

$$\boxed{\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)}$$

Therefore,

$$\boxed{\theta_j := \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} [(h_\theta(x_i) - y) x_i]}$$

Gradient descent works by moving downward toward the pits or valley in the graph to find the min value. This is achieved by taking the [illegible] derivative of cost function. Gradient descent is useful in optimization algo for linear regression, but it has some limitations and requires careful tuning of the learning rate to ensure [illegible] convergence.